

AMENDMENTS TO THE CLAIMS

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims

1. (Currently amended) A processing arrangement for a computer, the arrangement comprising:

a first processor means which is operable to process instructions from a first set of instructions; and

a second processor means which is operable to process instructions from a second set of instructions, which second set of instructions is a subset of the first set of instructions, the second processor means being arranged to receive instructions and to process the received instructions ~~without reference to~~ independently from the first processor means, when the received instructions are selected from the second set of instructions, wherein the first processor means includes a plurality of registers, and the second processor means is operable to access a predetermined, non-zero, selection of the said registers, and wherein the first and second processor[[s]] means are operable to process respective instructions in parallel with one another.

2. (Currently amended) An The processing arrangement ~~as claimed in of~~ claim 1, wherein the first processor means has active and inactive states of operation, and wherein the second processor means is operable to process instructions when the first processor means is in the inactive state.

3. (Currently amended) An The processing arrangement ~~as claimed in of~~ claim 2, wherein the second processor means is operable to cause the first processor means to change to the active state from the inactive state, when the received instructions cannot be processed by the second processor means.

4. (Currently amended) ~~An~~ The processing arrangement as claimed in of claim 3, further comprising a plurality of such processing arrangements having a plurality of second processor means and a plurality of corresponding first processor means wherein the each second processor means is operable to cause the ~~the~~its corresponding first processor means to change to the active state from the inactive state, when the received instructions cannot be processed by the its corresponding second processor means for processing respective subsets of the first instruction set.

5. (Currently amended) A method of operating a computer including first processor means which operates to process instructions from a first set of instructions, and second processor means which operates to process instructions from a second set of instructions, which second set of instructions is a subset of the first set of instructions, the method comprising:

using the second processor means to receive instructions; and

processing the received instructions using the second processor ~~without reference to~~ independently from the first processor means when the received instructions are selected from said second set of instructions, wherein the first processor means includes a plurality of registers, and the second processor means is operable to access a predetermined, non-zero, selection of the said registers, and wherein the first and second processor[~~s~~] means are operable to process respective instructions in parallel with one another.

6. (Currently amended) ~~[[A]] The method as claimed in of~~ claim 5, wherein the first processor means has active and inactive states of operation, and instructions are processed using the second processor means when the first processor means is in the inactive state of operation.

7. (New) A computer processing arrangement, the arrangement comprising:
a host processor, including a plurality of registers, operable to process instructions from a first set of instructions; and

second processor that operates to process instructions from a second set of instructions, the second set of instructions being a subset of the first set of instructions, the method comprising the steps of:

using the second processor to receive instructions; and

processing the received instructions using the second processor independently from the first processor when the received instructions are selected from said second set of instructions, wherein the first processor includes a plurality of registers, and the second processor is operable to access a predetermined, non-zero selection of the registers, and

wherein the second set of instructions used by the second processor is one selected from ~~one of a~~ the group consisting of servicing multiple tasks and exceptions, maintaining contexts of the first processor, and controlling interrupts received from peripherals coupled to the computer.

13. (Currently amended) The method of claim [[11]] 12, wherein the first processor has active and inactive states of operation, and instructions are processed using the second processor when the first processor is in the inactive state of operation.

14. (New) A computer system, comprising:

a host processor;

a shadow processor coupled to the host processor, the shadow processor adapted to control interrupts received from peripherals connected to a computer processor system;

the host processor in communication with an external bus via an external bus interface, the external bus adapted to transfer data to and from a main processor and at least one memory device;

a memory controller within the host processor for controlling data access to the at least one memory device;

an execution unit for controlling the memory controller and a main processor;

an arithmetic logic unit and a plurality of registers within the host processor;

the memory controller, arithmetic logic unit and registers and host processor being inter-communication via at least one internal bus;

the host processor adapted to process a first set of instructions;

the shadow processor adapted to process a second set of instructions, the second set of instructions being a subset of the first set of instructions; and

the shadow processor adapted to receive control signals and to process instructions in dependence upon those control signals independently of the host processor means.

15. (New) The computer system of Claim 14, in combination with a main processor.

16. (New) The computer system of Claim 15, further comprising the shadow processor being coupled to the main processor.

17. (New) The computer system of Claim 14, adapted to hold the host processor in a low power inactive mode while the shadow processor is processing a second set of instructions.

18. (New) The computer system of Claim 14, further comprising an interrupt controller within the shadow processor, adapted to receive interrupt signals via at least one interrupt input.

19. (New) The computer system of Claim 18, further comprising the interrupt controller being coupled to at least one peripheral via at least one peripheral bus.

20. (New) The computer system of Claim 14, the shadow processor further comprising registers which correspond to selected registers of the host processor.

21. (New) The computer system of Claim 20, wherein the shadow processor further includes an execution unit, code memory and data memory.

22. (New) The computer system of Claim 21, wherein the shadow processor is adapted to process a selected subset of the instructions from which the host processor operates, and these instructions are stored in the code memory.

23. (New) The computer system of Claim 22, wherein the shadow processor is adapted to access to the memory of the host processor by way of a memory controller which interfaces with the external bus by way of the host external bus interface.

24. (New) The computer system of Claim 23, further comprising:
a host interrupt controller adapted to couple the host processor execution unit to the shadow processor execution unit;
a register bridged unit adapted to couple the at least one register of the host processor to the at least one register of the shadow processor.

25. (New) The computer system of Claim 24, wherein the host interrupt controller is adapted to allow the shadow processor to issue an interrupt to the host processor in order to cause a change in task execution.

26. (New) The computer system of Claim 25 wherein the interrupt to the host processor uses a host interrupt protocol and indicates the source of the interrupt as a vector programmed by the shadow process related to the new task required.